Free Lunch?

Chris Saunders

University of Plymouth
MA/MSc/MRes Digital Art & Technology
Free Lunch?
Chris Saunders
October 2009

Acknowledgements

The author wishes to thank the following people:

David Clough, Nic Dacres, Gerry Heard, Alex Beston and Peter Lewis for their ideas, suggestions, time, unique skills and all round enthusiasm.

Mike Phillips for his feedback.

Geoff Cox for his patience.

Margaret Wells for her command of the English language.

Contents

Abstract

Last Christmas I gave you my heart but the very next day you gave it away - Wham

Giving a gift to someone establishes a relationship with the receiver and a debt. For as long as the receiver has not reciprocated, then the debt is outstanding. The act of giving can also show the personality of the giver and receiver and the depth of their current relationship. Superiority can be enforced by large displays of generosity. Gifts can have power and can improve status for the duration that they are in that individual's possession. But not all forms of reciprocation and debt are obvious. The debt can be with society as a whole or even nature. It can be cleared by a third party or someone or something that is not connected with the original benefactor at all.

Open Source Software has the ambiguity of being free. Whilst that does not strictly mean for non-payment, often the developers of the software give their time for free. But why do they? How can they work so enthusiastically on a project which does not give any financial return nor offer some other form of reciprocation?

This paper will examine the link between gifts and open source development. First it will investigate gifts and the psychology of gift giving. It will examine the bond created between giver and receiver. It will then focus on open source development and in particular UNIX/ Linux and Netscape Navigator/Firefox as examples. It will look at the collaborative nature of working and show how the donation of time and skills and the communities established were instrumental in their flourishing. It will then introduce the project free:dyne. The project is a portable operating system that enables users to carry their data and required software around on a portable USB device. By documenting the project, it will show how the methodology of open source development discussed earlier have been applied to the project.

Introduction

I'll give you television, I'll give you eyes of blue – David Bowie

Everyone likes to receive gifts. Birthdays, Christmas or maybe as a surprise and yet gift giving is fraught with danger. How do you handle receiving a gift with nothing to give in return? Or maybe you give and they have nothing for you? And what about receiving a collection of toiletries? Was that a subtle hint of an odour problem? As Aafke E. Komter explains in *Social Solidarity and the Gift* (2005), 'Gift giving is inherently risky[...] because of its psychological function of disclosing identities' (2005: 53). It also creates a tie between giver and receiver, 'people feel morally bound to one another because of the mutual expectations and obligations to return the gift' (2005: 43). It creates what Maurice Godelier refers to in *The Enigma of The Gift* (1999) as a 'twofold relationship between giver and receiver' (1999: 12). He further explains:

> 'A relationship of solidarity because the giver shares what he has, or what he is, with the receiver; and a relationship of superiority because the one who receives the gift and accepts it places himself in the debt of the one who has given it, thereby becoming indebted to the giver and to a certain extent becoming his “dependant”, at least for as long as he has not “given back” what he was given' (1999:12).

But are all gifts duty bound to be returned? The system of anonymous blood donation as exercised in the UK does not create a bond between the donor and the person receiving the blood nor does the donor expect the gift to be returned by the recipient. The same could be aimed at charitable donations. When one makes a pledge during a televised fund raiser, reciprocation is the last thing on the mind. So is that the case with open source software? Are programs and utilities freely given away by the developer without a need for some form of return?

Open Source Software evolved from the hackers of MIT who swapped programs between each other to help improve the efficiency of each other's code. As they believed that information should be allowed to move freely around a computer system, so they believed that all information should be free. This was something that Richard Stallman, who worked at MIT, embraced. As he is quoted in *Free Software, Free Society* (2002) 'If you

saw someone using an unfamiliar and interesting program, you could always ask to see the source code, so that you could read it, change it, or cannibalize parts of it to make a new program' (2002: 15). Following this belief is what frustrated him when he was refused permission when asked to see the source code of pre-compiled printer software. He saw this refusal as:

> 'a betrayal of the scientific mission that has nurtured software development since the end of World War II, it was a violation of the Golden Rule, the base line moral dictate to do unto others as you would have them do unto you' (2002: 12) .

Upon resigning from MIT, he set about developing a free suite of software products. To help him, he enlisted the help of fellow developers (hackers). This in turn later gave rise to the Free Software Foundation (FSF) which then helped to spawn the Open Source movement - this will be explained in greater detail later in this paper. Collaboration is key to open source development and this has been assisted by the rise of the internet. Members of a project development team no longer have to be in the same office, or even the same time zone come to that. Source code can be shared from a central repository with changes reported back to ensure a lack of conflict.

With people freely giving up their time, in a market economy this does suggest that someone somewhere is profiting on this donation. In her essay *Free Labor: Producing Culture for the Digital Economy* (2003), Tiziana Terranova suggests that the giving of an individual's time to a digital media project has essentially created 'NetSlaves'. She quotes an example of 'volunteers' asking the Department of Labor if AOL owed them back-pay for freely giving their time as acting as chathosts. This echoes McKenzie Wark's argument in *A Hacker Manifesto* (2004) where he describes hacking as being a class struggle against the owners of the systems who he has named 'the vectoralist class' (2004: 020). He equates hacking to the farming of land as both create an end product which is ultimately owned by other people. For the farmers it was the lords and barons and for hackers it's the owners of the intellectual property rights, the vectoralists. With this in mind, the AOL volunteers were freely giving up their time to help with AOL owned systems. They were essentially assisting AOL's Research and Development and indirectly aiding the advance of AOL's expansion and increased market share. They certainly appeared to have a case. Although their activities might have been 'pleasurably embraced' (2003), as Terranova points out, they were 'at the same time often shamelessly exploited' (2003).

However it appears that the digital economy overcomes this struggle. Terranova further writes,

> 'in the industrial economy the “worker tried to achieve fulfilment through leisure [and]... was alienated from the means of production which were owned and controlled by someone else,” in the digital economy the worker achieves fulfilment through work and finds in her brain her own, unalienated means of production' (2003).

Whilst payment may not have been in monetary terms, there did appear to be some form of reciprocation possibly more on an intellectual level. She further suggests that 'Such means of production need to be cultivated by encouraging the worker to participate in a culture of exchange' (2003). This would certainly reflect the community working practices of the hackers in the two examples discussed at length later in this paper. In each, ideas and suggestions were exchanged and in the majority of cases this would have been via the Internet effectively turning it into 'a channel through which “human intelligence” renews its capacity to produce' (2003).

This paper will examine the links between gifts, the gift economy and open source and with the examples given show that the methodology employed by open source developers is more productive than propriety software production and freely donating software is a better model of distribution than the current market economy model. It will also suggest that neither would be achievable without the sense of community and relationships formed as a result of the software project. First it will look at gifts and the psychology behind gifts. Using the work of Marcel Mauss as a starting point, it will look at the bond created between the giver and receiver and the solidarity it can create within society. It will also examine the conflicts between the gift of art and the need to make a living. The second chapter will focus on OSS development and distribution and in particular UNIX/Linux and Netscape Navigator/Firefox as examples. It will look at the collaborative nature of working and the important role that the internet has played. The third chapter will introduce the project 'free:dyne'. It will begin by describing the nature of the project, document its development and describe how it links to the argument of 'gift distribution' and community. The concluding chapter will summarise the main points raised to help demonstrate that the model used for OSS development and distribution is really only achieved by the community built around projects.

The Mouth of a Gift Horse

You can't give me the dreams they're mine anyway – Noel Gallagher

This chapter looks at the mechanisms of gift giving. By using examples such as the potlatch, blood donation, the Kula and the artistic gift, it will examine the reciprocation and the community bonds that are built through the act of giving.

As the saying goes, 'it is better to give than receive' which if that were true then why do we feel duty bound to give a gift in return? And what about a 'bad' gift? Why do teenagers feel aggrieved when they receive something that could be for a younger child? Or getting a pair of socks at Christmas, to roll out another well known phrase surely 'it's the thought that counts'. However, maybe the grievance is not with the gift itself but due to a lack of understanding of the receiver by the giver. The actual problem is that they did not really think about the receiver but just wanted to be seen as a generous person. As mentioned earlier, Komter states that gift giving reveals identities. With this in mind, it is possible that the inappropriate gift has revealed the true nature of the relationship between them. Either the giver does not truly know the receiver or that they do not really care enough to consider the nature of the gift. Either way, this clearly shows that giving a gift is not an act that should be entered into lightly for the repercussions could be disastrous.

The French anthropologist and sociologist, Marcel Mauss, explored the giving of gifts in primitive societies in his book *The Gift* (1990). In it, Mauss describes the obligations of gift-giving. The obligation of giving shows the giver as generous and is therefore deserving of respect. The obligation to accept because by receiving the gift that person shows respect to the giver and finally the obligation to return the gift to demonstrate that your honour is at least equivalent of the original giver. Gift-giving is therefore steeped in morality and by giving, receiving and returning gifts, a moral bond is created between the persons exchanging gifts. Mauss also emphasises the competitive and strategic characteristic of gift-giving. By giving more than your peers, you show greater respect than them and gift-giving contests, such as the potlatch, are common .

The potlatch is a ceremony practised among the North-West Coast Native American. At these gatherings a leader hosts guests in his home and holds a feast for them. The main

purpose of which is the re-distribution and reciprocity of wealth. Within it, hierarchical relations between clans and villages are reinforced through the distribution and sometimes the destruction of wealth and other related ceremonies. The status is raised by not who has the most but who distributes the most resources. The hosts demonstrate their wealth through the giving away of goods. A point further emphasised by Komter where he suggests that 'generosity and self-interest go hand in hand in gift exchange' (2005: 112). The objects exchanged in these 'primitive' societies, as Mauss pointed out, are laden with 'power'. Although a similar power is present to a certain extent in modern gifts, Mauss suggested that gifts in traditional societies are more complex than anything we know from modern society. The gift is more than a simple commodity or memento changing hands, it stands for every aspect of the society it belongs to. The gift is economic, political, legal, religious, magical, practical, personal and social. By moving such an object through the social landscape, the gift-giver is rearranging the fabric of sociality and it is this that forms the basis of the gift's power.

But what exactly is this power? As previously mentioned, Godelier believes that gift-giving builds a twofold relationship between the giver and receiver so is this power simply enforcing the feeling of superiority and the guilt to reciprocate? Komter seems to think that it is essential to maintaining solidarity between each other. He writes:

> 'Reciprocity appears to be the underlying principle behind gift exchange, with the connected feelings of gratitude functioning as the moral cement of human society and culture as such. Without gratitude there would be no social continuity as it fosters and maintains the network of social ties in which we are embedded' (2005: 58).

He further suggests:

> 'A sociological view on gratitude stresses the interpersonal relationships and social interactions in which gratitude takes shape. Gratitude is always embedded in a relationship between two parties. The capacity to be grateful and generous develops within the context of a social relationship. [..] Gratitude plays a crucial role in establishing and maintaining social relations' (2005: 67).

Certainly the act of giving can be seen as essential for maintaining a bond with our fellow man. At the end of the 20th century, donating and collecting for charity moved from religious institutions such as the Salvation Army to non-government organisations. Once it turned to the media it became more of an event echoing even the potlatch. Godelier writes:

> 'there is the appeal to outgive others, one city more than another, one company more than another, and there is the hope that the total will surpass that of preceding years. As in the potlatch, too, it is the practice to announce the names of individuals, towns, and companies who have shown the greatest generosity' (1999: 14).

He further suggests that the personal act of giving and the personal character of the gift is not diminished by the giver and receiver having no intimate, personal relationship. The relationship is formed by the virtual representatives shown on the program. He writes, 'children suffering from genetically transmitted diseases or AIDS victims are interviewed, and they arouse compassion and the desire to help, to give' (1999: 14). During Live Aid a video was shown of starving and diseased children in Ethiopia set to the song of 'Drive' by The Cars, the rate of donating was faster immediately after after its airing.

Clearly from the example above, the social bond that had been created had not been formed on the basis of reciprocation. Certainly not in the traditional sense as suggested by Mauss and Godelier above. The reciprocation is more than likely one of altruism with the giver experiencing a warm glow knowing that by donating they had helped another individual.

The same could be said of the donating of blood in the UK. Blood is collected by the National Blood Transfusion Service from volunteer donators. As documented in *The Gift Relationship* by Richard Titmuss (1997), in the United States of America blood donations are paid for. In studies performed by the Blood Services, Titmuss reports that 'the majority of donors are manual workers (employed and unemployed), particularly labourers, casual workers and semi-skilled operatives' (1997: 164). He goes on by saying that in these studies 'unemployed men (and particularly the young unemployed) were very heavily represented' (1997: 164). Certainly this cannot be good for the quality of blood collected. He quotes Professor J. Garrott Allen of the Stanford University School of Medicine as saying in 1966

> 'Blood from some groups among donor populations produces

> more cases of icteric hepatitis than blood from other groups. The incidence of icteric serum hepatitis in patients receiving single transfusions when the donors are of the prison-Skid-Row variety, is ten times that of volunteer donors, family or friends' (1997: 168).

Whilst paid donations with the flow of blood going from the poor to the rich could be considered morally distasteful, at least it can be argued that the poor are likely to be better off, financially at least. It is hoped that the dangers of donating will have been explained in particular the dangers of frequent donation. However, even being aware of the risks might not stop the donation. The need for money could be so great that it outweighs the risks forcing the donor to accept the consequences. Following this course of action could result in the act of donating causing a bigger sacrifice making him worse off even when the payment has been taken into account.

Apart from the morals surrounding the rich buying blood from the poor and the inherent dangers of multiple donations, the market driven system of collecting blood does not appear to be any worse than the volunteer system as performed in the UK. However, Julian Le Grand 's afterword in *The Gift* suggests otherwise. In it he proposes that under a market system, the risk of contamination is far greater. He explains:

> 'a commodity comes in two kinds: one that has the potential to do good (blood that has been uncontaminated) or harm (blood infected with a virus that is difficult to detect or to treat, such as hepatitis or AIDS). The supplier of the product knows which kind it is, but the potential buyer does not. [The] suppliers of the potentially harmful version of the product think only of their own interest: they wish to make the sale and are therefore prepared to sell it. [..] The harmful version of the product will therefore be sold along with the good version, with damaging consequences for the welfare of those who buy the former' (1997: 335).

He further suggests that:

> 'Under a voluntary system, the motives of donors are to benefit the recipients[..] the incentive here is[..] to reveal any information about the product that might harm the recipient.

> Hence the result of voluntarism is the provision of information so that only the good version of the product is provided. In the case of blood, only the uncontaminated kind is supplied' (1997: 336).

This is certainly a strong argument against a market economy and as Le Grand explains, it turns one of the traditional arguments in favour of markets, that of incentives, on its head and uses it to show the advantages of the voluntary system.

As with the Live Aid example earlier, reciprocation is not forthcoming. The donor does not expect, nor necessarily want, to meet the recipient and for them to return the blood donation nor would the recipient most likely be in a position to be able to. Although Komter rather cynically suggests that the main motive for donating blood is to be excused some other form of duty such as an afternoon away from military service. He also points out that the reciprocity could be postponed. He quotes one respondent as saying 'It can happen to me too, such an accident. You may be in need of blood yourself, at some time, and then you are lucky that there are some other people who have given their blood' (2005: 84). In a way, this is creating a gift-debt as suggested by Godelier. However, it is not between the two parties with the onus on the receiver to return, but more with society as a whole. A kind of insurance with the hope that someone else has been as generous; they are helping other people in the knowledge that they will be helped in return.

This form of indirect reciprocation is similar to the Kula exchange performed by the Massim peoples in Papua New Guinea. The area of participation covers eighteen islands with participants travelling hundreds of miles by canoe to perform the exchange. The gifts that lie at the heart of the Kula exchange are white shell armshells (considered 'female' and worn by men) and red shell necklaces ('male' and worn by women) and continually move around the ring of the islands, the armshells in an anti-clockwise direction and the necklaces clockwise. For example, if a person involved in the exchange is facing the centre of the circle, he will receive the armshells from his exchange partner on his left and pass them on to the partner on his right with the necklaces moving the other way. Although the gift must be passed on within a certain amount of time, ownership, no matter how temporary, brings with it status and prestige. Lewis Hyde in *The Gift* writes that 'to possess is to be great, and that wealth is the indispensable appanage of social rank and attribute of personal virtue' (2006: 15). Ownership also implies full ownership rights meaning that the

receiver can keep the gift, sell it or destroy it but an equivalent in value must be offered in its place likewise when offering a gift as part of the exchange. Hyde writes:

> 'the equivalence of the counter-gift is left to the giver, and it cannot be enforced by any kind of coercion. If a man gives a second-rate necklace in return for a fine set of armshells, people may talk but there is nothing anyone can do about it' (2006: 15).

It is the same with the duration of the gift in one's possession. Although duration with which a participant may hold onto a gift appears to be undetermined, as Hyde suggests 'if he keeps it too long he will begin to have a reputation for being “slow” and “hard” in the Kula' (2006: 14). What is important about the exchange is that it is not discussed. One may wonder what will be returned but will never bring it up. This is a very important distinction from bartering which is also practised amongst the islanders. With bartering the partners talk and talk until a deal has been struck whereas the gift is given in silence.

This form of circular giving is different from the simple method of reciprocal giving. When an armshell is given to the Kula partner on the right, an equivalent armshell is never returned. One has to give blindly in the hope that the returned gift will be to the recipient's liking. Hyde suggests that 'the circle is the structural equivalent of the prohibition on discussion' (2006: 16). However, with the gift effectively being passed on out of sight, it forces each participant to be a part of the group and each act of giving becomes one of social faith, much like the act of blood donation in the UK.

Similar to the Kula, the Maori of New Zealand have another example of circular gift giving but this extends beyond the tribe to include the environment. They have a word, hau, which translates as spirit and in particular the spirit of the gift and the spirit of the forest which provides them with food. When hunters return from the forest with birds they have killed, they give some of the kill to the priests who then cook the birds over a sacred fire. The priests then eat them and from the remains make a talisman called a mauri which acts as the physical incarnation of the hau. In a ceremony called whangai hau, meaning 'nourishing hau', the mauri is given to the forest as a way of ensuring the birds stay abundant for the hunters to continue to kill and to provide as food. In this ritual there are three gifts; the forest to the hunters, the hunters to the priests and the priests back to the forest. By involving the priests, it is removing the possibility of the exchange resorting to barter between the forest and the hunters with the latter potentially looking to exploit the

situation and make a profit – at a possible detriment to the number of birds available to hunt. With the priests involved, the gift leaves the sight of the hunters before it is returned to the forest. Although it is plain to see that the mauri does not really cause the birds to be abundant, but performing this ritual includes man in the wider ecological cycle and it can be seen that what nature gives man is influenced by what man gives back to nature. By not being greedy or exploiting the forest, the birds' abundance is a direct consequence of man treating the forest's wealth as a gift.

What can be seen from the examples so far is the power of the gift is not diminished once it has been given or used. In fact it could be argued that when a gift has been used it is not used up but the opposite is the case; a gift that is not used is lost. Hyde explains,

> 'Gifts are a class of property whose value lies only in their use and which literally cease to exists as gifts if they are not constantly consumed. When gifts are sold, they change their nature as much as water changes when it freezes, and no rationalist telling of the constant elemental structure can replace the feeling that is lost' (2006: 21).

Parallels can be drawn between OSS developers and artists with both needing to earn in order to survive and yet still stay true to their creative ideals. When we speak of talent or inspiration we often refer to it as a gift. When an artist is working, an idea or a tune or a phrase will come into his head or as Hyde explains it, it is 'bestowed upon him' (2006: xiv). Much like a tangible gift, this gift is not something that we can buy or barter for rather it is something that is acquired without any effort on the benefactor's part. Certainly lessons can be purchased to hone the talent but no will in the world will induce its initial appearance. Hyde suggests that a work of art is a gift and not a commodity but it can live in both the gift and market economy. However, although it can exist without the market, without the gift there would be no art. With that in mind, if art is a gift, how does an artist reconcile his need to create his art and still survive in a society dominated by a market economy? He needs to convert his gift-wealth into market-wealth in order to live and to be able to support his art. Hyde believes that there are essentially three ways in which the modern artist chooses to resolve this dilemma; they have a second job, they have a patron to support them or they sell the work of art on the market themselves. Taking a second job allows the artist to separate his gift from the market economy. Typically the second job will

be far removed from the artist's gift as if to emphasise this division more such as a painter working as a data input clerk. As Hyde writes, 'The second job frees his art from the burden of financial responsibility so that when he is creating the work he may turn from questions of market value and labor[...] He earns a wage in the marketplace and gives it to his art' (2006: 278). An artist with a patron (or more than likely these days with a grant) is more detached from the market. Although the money received from a patron might seem like a wage or a fee for a service, it is a gift made in recognition of the artist's own gift allowing him the freedom to create. As can be seen with this method of financing, there is a distinct division of roles. The artist remains in his 'gift zone' whereas the patron has at some time or other had to enter the marketplace to acquire his wealth to turn it into a gift for the artist.

From the two examples above, it is easy to see the distinct boundary between the artist's work and the market, but what of the artist who must sell his own work? Ideally he will need to be able to have a subjective opinion of his own work; be able to withdraw from it and to think of it as a commodity. He will need to be able to value it in terms of current trends, how the market is performing and be able to part with it when it has been sold. In terms of his art, he must be able to turn his back on the marketplace and work without it as a distraction. As Hyde predicts:

> 'If he cannot do the former, he cannot hope to sell his art, and if he cannot do the latter, he may have no art to sell, or only a commercial art, work that has been created in response to the demands of the market, not in response to the demands of the gift' (2006: 279).

If an artist hopes to sell work that has been produced as a result of his gift and not due to fashion or trends resulting from market forces, then he must totally ignore the market and create for himself. Once he is happy that his work is a true realisation of his gift he can then see if it has any value in the market economy. As way of illustration, Hyde discusses the work of the American realist painter Edward Hopper. During the early years of his career, Hopper was a commercial artist designing covers for trade magazines. As Hyde explains, 'they are not art. They certainly have none of Hopper's particular gift[...] into the way that incandescent light shapes an American city at night' (2006: 279). They were simply in reply to the demands of the market. During this time however, Hopper still created for himself. Although managing to sell a painting in 1913, he did not sell another

until ten years later. Hopper's magazine work ought to be considered as a second job and not as a part of his art. Certainly the 'happy couples in yellow sailboats and businessman strolling the golf links' (2006: 280) could not in any way be compared to his work entitled Nighthawks, a painting of a New York diner at night with its fluorescent lights breaking up the outside gloom. Even when market demands dictated one form of art, Hopper stayed true to his gift. He could have made a comfortable living designing magazine covers but instead he saw this work as a way to support his true calling. The distinction between market driven art and gift inspired art might be blurred at times but it is one that artists must be able to reconcile in order to maintain their integrity.

This chapter has looked at the psychology of gift giving and the bond that is created between giver and receiver. By use of examples such as the potlatch, blood donation and the Kula, it has examined the idea of reciprocity and has shown that it does not need to be returned directly to the giver. It has also examined the tension between the gift of art and the artist's need to earn money to both survive and to feed his gift. The next chapter will look at the origins of Open Source Software, the collaborative nature of development and its distribution.

Geeks bearing Gifts

Take this letter that I give you, Take it sonny, hold it high - Queen

The previous chapter examined gift giving, the psychology of giving and reciprocation and the social bond that is created. An important aspect of Open Source Software (OSS) is the donation of code and time to projects as a gift and the communities built to aid development and provide support. This chapter will look at the history behind OSS and how the 'gift' methodology came about. Using the examples of Linux/UNIX and Netscape/ Firefox show that this model would not be a more efficient method of development and distribution than the current marketplace driven model were it not for its community basis.

The origins of OSS development can be traced back to the MIT Tech Model Railway Club of the late 1950s. Within it was a sub-group called the Signals and Power Subcommittee who were constantly coming up with ways of controlling different trains on different parts of the track using old equipment. As important as these improvements were, what was equally important was the manner in which it was performed. Steven Levy in *Hackers* (2001) explains:

> 'While someone might call a clever connection between relays a "mere hack", it would be understood that, to qualify as a hack, the feat must be imbued with innovation, style and technical virtuosity[...] the artistry with which one hacked was recognized to be considerable' (2001: 23).

From manipulating, or hacking, train sets, the subcommittee moved onto computers when in 1961 MIT acquired the first PDP-1. As Eric S. Raymond explains in *The Cathedral and the Bazaar* (2001) '[they] adopted the machine as their favourite tech-toy and invented programming tools, slang and an entire surrounding culture' (2001: 4). Levy refers to this culture as 'The Hacker Ethic' (2001: 39-49) which had six basic ideals:

1. Access to computers – and anything which might teach you something about the way the world works – should be unlimited and total. Always yield to the Hands-On Imperative.
2. All information should be free
3. Mistrust Authority – Promote Decentralization
4. Hackers should be judged by their hacking, not bogus criteria such as degrees, age, race,

or positions.
5. You can create art and beauty on a computer
6. Computers can change your life for the better
Whilst throughout this paper there will be examples of some of these ideals, the one that is at the core of the ethic and OSS is the belief that information should be free.

However, during the 1970s a gradual change was taking place that would challenge the notion of free and make more of a formal distinction between 'marketplace' software and OSS. There was a move away from the understanding that software was something that could be shared, adapted and used to enable users to understand their machines better to one of a more commercially driven approach. This change has often been dated back to Bill Gates' Open Letter to Hobbyists published in the Homebrew Computer Club newsletter in February 1976. In it he berates the hackers from stealing his company's software. Freely sharing and not paying, Gates believed, was detrimental to the advance of good software. As he wrote, 'Who can afford to do professional work for nothing? What hobbyist can put 3-man years into programming, finding all bugs, documenting his product and distribute for free?' A dilemma similar to the one artists have been struggling with for years as discussed in the previous chapter. This commercial approach has often been used to vilify Microsoft and the proprietary nature of their products. However, as will be seen later sharing has actually been beneficial to the production of good, robust software that has even challenged commercially-built products.

It is hard to pinpoint what the influences were behind the ideal that information should be free. The counter-culture of the 1960s must have had a bearing. Certainly these hackers would have been influenced by the computers on which they learnt their skills. Openly sharing resources such as memory and not keeping it locked would have improved the efficiency of the computer. In the 1960s, computers had almost no systems software at all. So to reduce time-wasting and effort in developing different versions of the same utility, they had a drawer next to the system console containing tools available to anyone who needed it. Swapping programs and helping each other improve the efficiency of their code was accepted as the norm. Being generous with code and assistance improved a hacker's standing amongst his peers, much like the generosity of the host during a potlatch and no doubt an element of self-interest was not unforthcoming. However, what would have had a large influence, and can be seen to have parallels with OSS development, is the academic

environment in which they were immersed. Scientists openly make their work available for others to explore and further develop. Pekka Himanen in *The Hacker Ethic* (2001) explains 'The scientific ethic entails a model in which theories are developed collectively and their flaws are perceived and gradually removed by means of criticism provided by the entire scientific community' (2001: 68). It all starts with a problem that a researcher finds interesting and for which they might also have discovered a solution. The ethic dictates that this is made available for anyone to use, criticise or further develop. By opening up the research to the community, scientists can get valuable feedback. As important as the solution is the work that produced it. As Himanen suggests, 'It is not enough to merely publish '$E = mc^2$' – theoretical and empirical justifications are also required' (2001: 69). But it is not just about rights, sources must always be included as plagiarism is severely frowned upon and any new solutions must also be published to the community. Although both of these requirements are not required by law, there is a moral duty to conform enforced by the scientific community. The opposite of this as suggested by Himanen is one which can be called 'the closed model' (2001: 70) in which it is not just information that is closed off and restricted but is also authoritarian. In this model, some authority chooses the research goal and creates a closed group of people to develop and implement it. As in accordance with the hacker's third ideal, scientists avoid formal hierarchical structures. Without them, they are free to explore their passions and interests and can network and work with other people who share them. But this does not mean that there is a state of anarchy, projects have their figureheads who decide on direction and support others as well as a formal publication structure. Although anyone can perform research, only a select number of projects are included in reputable scientific journals and these are decided upon by a smaller group. This group retains its position, 'only as long as its choices correspond to the considered choices of the entire peer community' (2001: 72). If it is unable to do so, then the group is bypassed and another channel is created, as Himanen writes 'it is the truth that determines the[...] group' (2001: 72).

To examine this methodology further in terms of software development, lets look at the evolution of UNIX and subsequently Linux. UNIX was initially developed in 1969 by Ken Thompson and Dennis Ritchie and was the product of a project called Multics[1] worked on by MIT and funded in part by Bell Labs. Thompson had been assigned to the project by Bell and when they pulled their support for the project, he went back to work solely for

1 Multics was an early time-sharing operating system which allowed multiple users on one computer at the same time.

them. Bell Labs allowed their researchers to work on anything as long as it had some relation to the company's interests much like work performed by hackers for the vectoralists. So over a period of two years, using discarded equipment, Thompson and Ritchie developed an operating system (OS) including the programming language C and a host of tools. The OS was christened UNIX - a pun on Multics. The freedom to follow their own path was something of a double-edged sword as it meant that there was a complete lack of funding from Bell. However, this did not seem to inhibit development, as Christopher M Kelty reports in *Two Bits*, 'it influenced the design of the system, which was oriented toward a super-slim control unit (a kernel) that governed the basic operation of the machine and an expandable suite of small, independent tools' (2008: 126). Although proprietary[2], UNIX was not commercialised or marketed, instead for a small licensing fee, AT&T allowed anyone (from individuals up to corporations) to install it and to create derivatives. This was achieved due to the fact that as part of the licence a tape containing documentation, a binary version of the software and the corresponding source code for it was included. Like the Kula with full ownership rights over the gift before passing it on, having the source code encouraged users to maintain and extend it and to funnel the changes back to Thompson and Ritchie. As can be seen, this also mirrors the aforementioned scientific method with the release tape acting as the means of making research available to be further developed and Thompson and Ritchie as the 'publishing' group. Kelty further writes,

> '[people] found it appealing, if not more, to be involved in the creation and improvement of a cutting-edge system by licensing and porting the software themselves[...] The means by which source code was shared, and the norms and practices of sharing, porting, forking, and modifying source code were developed in this period as part of the development of UNIX itself' (2008: 128).

Throughout the 1970s UNIX was ported to a number of systems and its use had spread around the globe with communities, or user groups, being formed in Canada, Europe, Australia and Japan. New tools and applications were developed by these communities and circulated independently as well as being included in the frequent releases from Bell Labs. This raised some interesting questions. Were all the fixed, ported and extended versions

2 UNIX was copyrighted and wholly owned by Bell Labs who was ultimately owned by AT&T.

still UNIX? Who legally owned the code contributed by non-Bell employees? Was it still under the control of AT&T or were the changes significant enough for it to be considered as something else? Whilst these were all relevant questions, something far more interesting was unfolding. As Kelty explains, 'just about every possible modification has been made, legally and technically, but the concept of UNIX has remained remarkably stable' (2008: 131).

UNIX became an important teaching aid for academics due to it being the ideal concept of a time-sharing, multi-user operating system. Andrew Tanenbaum, a computer scientist at Vrije Universiteit in Amsterdam, had used UNIX in his classes but when AT&T released version 7 the new licence stopped the source code from being studied. Wanting to continue teaching UNIX but also not wanting to break the law, he developed his own version without a single line of AT&T code called Minix. This was a stripped down version for running on PCs and was distributed along with a textbook called Operating Systems published by Prentice Hall. Minix was widely used as a teaching tool in the 1980s and although Tanenbaum received suggestions for changes, few were included. Partly due to his teaching commitments and partly due to his wanting to keep it simple enough to be understood by students and printed in a textbook. Ironically, this reluctance to extend Minix could have been the starting point for possibly the most significant and the epitome of open source projects to date.

In 1990, Linus Torvalds, a student at the University of Helsinki, was in the audience of a talk given by Richard Stallman. Although he did not agree to Stallman's socio-political agenda, as Sam Williams in *Free as in Freedom* (2002) recounts,

> 'Torvalds[...] appreciated the agenda's underlying logic: no programmer writes error-free code. By sharing software, hackers put a program's improvement ahead of individual motivations such as greed or ego protection' (2002: 136).

Unlike Stallman who came up against bureaucracy, Torvalds' barriers had been geography and the harsh Scandinavian winter. Forced to walk across the campus to logon to the UNIX system made him go in search for a method that would allow him to access it from the warmth of his room. His search led him to Minix. Whilst it would run on a 386 PC, it lacked a few necessary features such as a terminal emulator that would enable him to remotely access the university's system. So, as with Tanenbaum creating his own version of

UNIX, in the summer of 1991 Torvalds wrote his own version of Minix from the bottom up. He got a copy of the POSIX[3] standards from a Minix newsgroup and a few week later, as Williams reports, he posted the following message on the comp.os.minix Usenet group with the title “What would you like to see most in minix?” (2001: 64):

> 'Hello everybody out there using minix - I’m doing a (free) operating system (just a hobby, won’t be big and professional like gnu for 386 (486) AT clones). This has been brewing since April, and is starting to get ready. I’d like any feedback on things people like/dislike in minx, as my OS resembles it somewhat (same physical layout of the file-system (due to practical reasons) among other things)' (2002: 137).

Torvalds received some ideas and offers of help in testing in reply. The first release of Linux[4] was in September 1991 as free source code with the next, improved version in early October of the same year. The important aspect of Torvalds' work was the community he created by opening up the project to others. As Himanen writes, 'Thousands of programmers have participated in Linux's development, and their numbers are growing steadily. [...] Anyone can participate in its development, and anyone is welcome to use it freely' (2001: 65). Much like the donation of blood in the UK, the work performed did not appear to expect any form of payment or reciprocation. At least not in the traditional sense. With such a large workforce it is a wonder how the workload is divided up. Charles Leadbetter has the answer in *We-Think* (2009) where he writes

> 'They motivate a mass of contributors by providing interesting work, posing interesting questions and attracting interesting people to work with. The[...] performance is judged by your peers, and the community shares an overarching goal. Authority is exercised mainly by peer review and with a light touch. The traditional organisation co-ordinates work through a division of labour: people are allotted tasks divided up from above, on the assumption that people who design the process or production line know what needs to be done. Open-source communities co-ordinate people as they innovate. People distribute themselves to the tasks they think need doing and for which they have the

3 POSIX is the blue prints that determines whether a program is UNIX compatible or not.

4 Linux was the name that Torvalds had used for the program on his hard drive following the convention that every UNIX variant's name had to end in a letter X. Deciding that it was too egotistical, he changed it to Freix but was persuaded by the FTP site manager to rename it back.

> skills. A self-distribution of labour[...] is far cheaper and more innovative than a centrally planned division of labour'
> (2009: 110).

But how did the project manage to attract such great numbers of developers? Something about Linux fired the imagination of the hacker community. Douglas Thomas in *Hacker Culture* (2002) offers as way of explanation that 'it was, in essence, the perfect hacker project, requiring the user to learn about his or her machine at just about every turn' (2002: 84). 'The hacker is[...] enthusiastic about this interesting thing; it energizes him' writes Himanen (2001: 4) and that is the key. Working on aspects of the project that interest the developer helps to maintain the desire. The gift-debt of the donation of their time is reciprocated by intellectual stimulation. With an enthusiastic workforce at his disposal Torvalds still needed to co-ordinate development and control the versions released. Work was divided into modules with hacker groups developing competing versions. Much like the publishing group in the scientific method, there is a small group of Torvalds and a few principal developers who decide which of these versions will be included in the improved version of Linux. As in the potlatch, each development group would be trying to out-do the other with their code. However, Torvalds' group only retains its authority as long as its decisions match the general consensus of the community as a whole. To control the OS continuous development, releases are numbered in the format x.y.z. For stable versions, suitable for the average user, the y is an even number but versions aimed at developers, the y is the stable version plus one. For example, version 1.0.0 would be a stable version but version 1.1.0 would be an untested improved version. X would only increase when a fundamental change is made. Himanen writes, 'This simple model has worked surprisingly well in the management of Linux development' (2001: 66).

Having a highly motivated workforce developing self-chosen problems, peer review as quality control and simple methods of release it is no wonder that in a short space of time as Raymond reports, 'Linux could compete on stability and reliability with many commercial UNIXes and hosted vastly more software' (2001: 16). According to Leadbetter 'by 2006 Linux accounted for about 80 per cent of software on computer servers worldwide' (2009: 66). As its penetration grew, so did its complexity. Leadbetter further writes:

> 'The version released by Debian in June 2005 had 229 million

> lines in its source code and would have taken 60,000 person-years of software-coding, at a cost of perhaps $8 billion. (To put that in context, the version of Microsoft Windows XP released in 2002 had an estimated 40 million lines in its source code.)' (2009: 66).

No matter how inconsequential a hacker's work on the project was, their contributions helped to make Linux what it is today. Just because their work has been included does not mean that the power of their assistance has diminished. Its value, just like that of a gift, lies in its use.

From a purely economic point of view, the methodology used by OSS projects seems to make perfect sense. Simply open the project up to the hacker community, let them work on problems that they find challenging and interesting and a robust product will be the result. However, freely giving away your company's 'secrets' might not go down to well with its investors. This was a problem that Netscape faced.

In 1993 a student called Marc Andreessen, along with Eric Bina and other hackers, created a user-friendly graphical browser for the PC at the National Centre for Supercomputing Applications (NCSA). Originally called Mosaic, it was free for non-commercial use on a multitude of platforms including the Commodore Amiga. The source code was publicly provided for the UNIX version although interestingly it was never released as OSS. The development team left NCSA to form Mosaic Communications Corporation which later became Netscape Communications Corporation following legal threats from the NCSA and a rival firm. Kelty reports that 'Netscape was known as home to a number of Free Software hackers and advocates, most notably Jamie Zawinski' (2008: 100). It was Zawinski who would head the new free browser project called Mozilla, later Navigator.

Access to the World Wide Web has always meant to be platform independent which in practice meant that browser developers had to port[5] the code over to different computers. Navigator was available on many computer architectures already but by 1997, Netscape were already investigating a Java version. Created by Sun Microsystems, the advantage of Java-developed software is that it can run on any platform. Sun even coined the slogan 'write once, run everywhere' to help market the language. Whilst this would appear to solve

5 Porting is adapting software so that an executable program can be created for a computer or OS differing to the one for which it was originally designed.

the problem of platform independence, it also introduced another issue. To run a Java program, bytecode[6] would need to be created which in turn would be transmitted from the server to the computer executing it. Anyone knowing where to look would be able to reverse-engineer this code. Although obfuscating the code would be an option, Zawinski was against this and suggested sharing the code and getting other people to help improve it. Kelty reports, 'As a longtime participant in Free Software, Zawinski understood the potential benefits of receiving help from a huge pool of potential contributors' (2008: 102). Whilst this would be a sensible move from a software engineering point of view, it was far from an intelligent business plan. As Kelty further explains, 'such a move seemed tantamount to simply giving away the intellectual property of the company itself' (2008: 102). And yet, Netscape had already set a precedent with donating software for free. It had already made Navigator freely available to download which was in direct opposition to the more recognised practices of commercial software. This can be seen as instrumental in Netscape's accelerated growth. Similar to the three gift hau exchange being responsible for the abundance of the birds, the free donation of the browser improved Netscape's value and visibility; the browser was downloaded, shares were bought, profile raised. To further endorse Zawinski's aim, a sales manager called Frank Hecker wrote a document circulated to senior management in which he made the case for releasing the source code as a Netscape product. So, in January 1998, Netscape issued a press release stating that they would make available the source code of the next version of their browser, Communicator.

At the same time as Netscape's decision, interesting events were taking place in the hacker community. Since founding the FSF, Richard Stallman had been working towards a free UNIX-compatible system called GNU (Gnu's Not UNIX). He started by creating tools for the system with the first being a text editor called Emacs. During this time, he encountered software copyright. Whilst initially being against this, he changed his opinion when he read 'verbatim copying permitted' in an email. Using this as inspiration, Stallman developed the GNU Public License (GPL) which gave users permission to run, copy, modify and redistribute the software providing they did not add any restrictions of their own. Despite the work being produced, there were cracks appearing. Stallman liked to micro-manage all development work which frustrated people and resulted in a mass exodus of staff from the FSF including Eric Raymond. Raymond liked Torvalds' relaxed style of project management with Linux and adopted it with his work on 'fetchmail', a popmail utility.

6 Bytecode consists of compact numeric code which is executed one instruction at a time but requires compiling into machine code at runtime. Whilst the compiler would be computer specific, the bytecode would not need to be.

After opening the project up, he was soon receiving bug reports and enhancement suggestions with the resulting software proving to be quite robust. He wrote his observations along with a comparison of the GNU and Linux management styles in a paper which he later presented at conferences. In the audience of one such conference were representatives from Netscape who invited him to their offices to offer advice. Whilst there he sat in on a meeting to discuss how other corporations could follow Netscape's lead. One of the points discussed was the name 'free'. Most executives would associate the term with zero cost despite Stallman and other hackers original definition explaining it that it was as in free speech and not as in free beer. During this meeting the term 'open source' was mentioned. Not initially accepted as official, it was quickly adopted although not by Stallman who still saw non-free software more as a social problem rather than one of practicalities.

Tim Berners-Lee, the 'inventor' of the World Wide Web, always believed that the protocol and all of its implementations should be made freely available. Releasing the source code could be seen as a way of appeasing people such as him. But during the 1990s, both Netscape and Microsoft (with Internet Explorer) had strayed from this. As Kelty writes, 'each had implemented its own extensions and “features”[...], extensions not present in the protocol that Berners-Lee had created or in the subsequent standards created by the World Wide Web Consortium (W3C)' (2008: 103). So, an Open Source browser would be a way to circumnavigate these non-standard browsers by making it comply with standards. Unfortunately there laid a problem. Instead of utilising existing licences like the GNU GPL, Netscape created their own called the Netscape Public License (NPL) and the Mozilla Public License. Netscape's problem was they might not have the rights to redistribute third-party developed code as Free Software. Existing licences either gave other people rights that they themselves did not have or were too restrictive. To overcome this, they wrote two licences; one for existing code (NPL) and the other for new contributions. This dual licensing system did nothing to endear Netscape with the hacker community including Richard Stallman who urged people not to use the NPL. Bruce Perens who created the Open Source Definition is quoted by Kelty as saying

> 'the NPL was designed for the specific business situation that Netscape was in at the time it was written, and is not necessarily appropriate for others to use. It should remain the license of Netscape and Mozilla, and others should use the GPL or the

BSD or X licenses' (2008: 105).

Releasing the source code meant that a system for co-ordinating the work of developers outside of Netscape was needed, as was a method to make them want to contribute and to enable them to see if their code was included or rejected. To facilitate this the domain mozilla.org was obtained and tools were developed to control all of this effort such as a system for tracking bugs reported by users and developers (Bugzilla). These tools also had their source code released . With everything in place, the intention was for paid developers and editors to check and verify whether the contributions could be included. But, the project was still seen as a Netscape project due to it being made up of Netscape employees. In Zawinski's opinion, the project was a failure. In his letter of resignation he described the project as too large and the code too complicated for outsiders to be able to just jump straight in and start making changes. He believed that Netscape had declined because it was no longer innovative and had become too large to be creative. By the releasing of the source code, it was a way for it to return to its former glory as well as proving the power of OSS. Kelty quotes Zawinski further:

> 'I saw it as a chance for the code to prosper[...] the fact that Netscape was no longer capable of building products wouldn't matter: the outsiders would show Netscape how it's done. By putting control of the web browser into the hands of anyone who cared to step up to the task, we would ensure that those people would keep it going, out of their own self-interest'(2008: 107).

Unfortunately, this failed to entice people to take up the challenge or at least not at the speed that he or other people in the hacker community had imagined. By 2002 over 95% (www.onestat.com) of users on the Internet were using Internet Explorer in one form or another. In that same year, Phoenix (later renamed Firefox) was released to a largely un-noticing public. Within six years of that release, as well as offering users a choice of browser, it was setting records including the most downloaded software in a 24 hour period with over eight million people downloading version three. Zawinski's dream had finally been realised. Although he did not know it at the time, the words Zawinski used in his resignation back in 1999 were quite prophetic:

> 'merely by being who we are and doing what we did, we played a big part in bringing the whole open source development model

> to the attention of the world at large. We didn't start the mainstream media interest in open source (Linux did that, mostly), but I think we did legitimize it in the eyes of a lot of people, and we did tell the story very well. Lending the Netscape name to this software development strategy brought it to the attention of people who might otherwise have dismissed it' (1999 www.jwz.org/gruntle/nomo.html).

At least eight million people in 2008 appeared to have agreed with him.

This chapter has looked at the nature of the OSS methodology using Linux and Netscape/Firefox as examples and has suggested how it is similar to that of scientific research. The free giving of time and skills by hackers to a project is similar to the system of blood donation in the UK and making contributions to charity. Whilst reciprocation is not immediately apparent in the obvious sense, it does build a social bond, in the case of OSS, with the community of fellow hackers working on the project. Each competing with each other for the right to have their code included in the next release. The next chapter will document the free:dyne project and show how it uses OSS and the community-based method of development as references.

free:dyne

I want it now give me your heart and your soul - Muse

In the previous chapter, OSS, the gift methodology and the community form of development was explored. This chapter will document the project free:dyne and explore how it has followed the concepts of the development and distribution methodology discussed earlier.

free:dyne was inspired by two existing 'portable' systems - dyne:bolic and pure:dyne – and maintains some of the ideals that they initiated such as developed on and with OSS tools, their portability and being freely distributed. dyne:bolic is a multimedia system containing tools to allow media artists to manipulate audio and video with tools to record, edit and stream. Customisation is possible via additional modules and can be achieved by simply downloading them into the modules directory. However, it is recommended to only use authorised modules. Creation of the system is simply achieved by downloading an ISO file from the website and then burning it to CD. From there, a version of the system can then be transferred onto a USB storage device. Initially created by Jaromil, there is a community of developers who help with extending its functionality as well as online support in the way of a website and documentation. pure:dyne is similar to dyne:bolic in that it too is a multimedia system containing tools for audio and video manipulation. Creation is also achieved by downloading an ISO file from the website and burning it to CD and like dyne:bolic, it can be transferred onto a USB storage device.

From the outset of the project, there were certain criteria free:dyne had to meet. First were the tools included and base operating system that would be used. As with the other two systems, it was important they all be OSS. Second was the use of community to develop it. As can be seen in the previous chapter, opening up a project to a community of developers for assistance and quality control makes for a robust product. Thirdly was ease of modification and construction. Whilst it is possible to adapt the other two systems, dyne:bolic prefers the user to install 'approved' modules and both have quite a complicated method of transferring onto a USB device. As the main users of both of these systems appear to be software artists of varying levels of technical expertise, it would appear that they would be equipped to perform this procedure. free:dyne however is to be targeted at

the less-savvy user so the method employed would need to be fairly easy. Finally was its portability. The host from which it was booted would only be needed for its processing power, in other words the user would be able to carry their data around with them on the same device, again, like the other two systems. With questions being raised about cloud computing, it was important from the outset that the user should maintain full ownership and rights to their data.

Cloud computing appears to be the latest remedy to the ever increasing requirements of organisations and individuals alike. Essentially cloud computing allows the user to avoid expenditure on hardware, software and services by 'renting' what they need when they need it. The word 'cloud' in this context is a metaphor for the Internet as this is the medium over which the services are delivered, generally through a web browser. However, what appears to be a flexible, cost effective solution opens up another area of discussion. As Jonathan Webber pointed out in The Times (05 May 2008) the user is once again dependent on someone else for their technology as they were in previous years when connectivity was via dumb terminals[7]. They can only do what the central administrators have granted authorisation for them to do. Nor do the users physically possess their data as it is stored centrally on the provider's server. The responsibility for ensuring adequate backups in the event of a situation requiring some form of recovery notwithstanding, access to the data is again in the hands of the central administrators. Users appear to be sacrificing their private and personal data to a third party for a reduction in monetary outlay.

The first decision to be made was with the base OS. Obviously it would need to be Linux but with so many different versions available, it was important to chose one that not only would support running from a USB device but would be easy to use and offer the smallest learning curve in the transition from Windows. An email was sent out (see Appendix 1) to the mailing list for Freelancers.net asking for advice on the best Linux distro[8] that would fit the above criteria. Six responses were received with them all suggesting Ubuntu 9.04 although one also suggested looking at openSUSE. Ubuntu has a inbuilt feature for creating a version on a USB drive whereas openSUSE's method was similar to dyne:bolic and pure:dyne so it was decided to use Ubuntu. Gnome was suggested as probably the

7 A dumb terminal is a device that cannot process or store data locally nor execute user programs.
8 A distro is a set of software components that have been packaged into a larger product or component for distribution to end-users.

easiest interface to follow and understand for Windows users. Although this desktop environment would run on any version of Linux, due to the simplicity mentioned above Ubuntu was chosen. Of the six respondents, five agreed to provide support in the form of suggestions, development and testing.

The second decision that needed to be made was on the applications to be included and the target device. Following discussions with the support team a list was drawn up (see Appendix 2). Ubuntu already has a number of useful tools pre-installed such as a video editor and Open Office so it was decided to take advantage of those. During the discussions a number of alternatives were suggested and so initially these were all included. The reason for their exclusion is discussed later. But, as the purpose of the OS was to be aimed at as large a number of people as possible and removing some of the applications could inadvertently alienate some users, it was agreed that there should be a number of different versions catering for different needs. Starting with a full version, one for users interested in audio/visual work, one for graphic designers, one for developers and a base version to allow people to make their own distributions. The decision for a base version came out of a suggestion of one of the support team that free:dyne could become a generic name for a family of portable systems, each with their own unique features. For the target USB device, a 4GB USB stick was chosen. Although it would be possible to install free:dyne onto any USB storage, research showed that the entry level size appeared to be 4GB hence the decision to aim for that size.

The final decision to be made was on the promotion of the OS. Although it would have been attractive to attach it to an existing organisation, following on from the decision to aim at making free:dyne a family of distros, the team felt that it should have its own identity. Whilst it was hoped that it would eventually have 'a life of its own', all members of the support team felt that there should be some link with the author. Consequently, the corresponding website with documentation and distros to download was constructed under a subdomain of the author's existing site and the URL for the OS became freedyne.boxel.co.uk. Wordpress was chosen as the base platform for it due to its ease of deployment and also because it is open source. To help with the promotion of the OS, a workshop was planned. There will be a fuller description of this later.

To create a platform from which all of the initial distros would be created, a full version of

Ubuntu 9.04 was installed onto a PC making it dual boot into Ubuntu and Windows. A PC was chosen as the development machine as it was felt the majority of users would more than likely have access to one than any other platform. Once this was completed, each of the chosen applications was installed. It was important to make the steps to create the portable OS as simple as possible and following discussions in the support team, it was decided that it would have to be a two step process. The user would download a disk image (ISO) file, burn it to DVD to create a LiveDVD and then use Ubuntu's inbuilt 'USB Startup Disk Creator'. Whilst this was not ideal, it was by far and wide the simplest for a non-technical user to perform. To create the ISO file which would make up the distro, Remastersys was used as this would allow the end user to create a LiveDVD from it which in turn was needed to perform the final step of writing to a USB stick. Full instructions of how to create this are on the website under documentation (i.e. freedyne.boxel.co.uk/documentation). The first ISO file made was just under 3.5GB. Whilst it would have fitted onto the target device, there would not have been much available free space for the user's own data. It was this that prompted the trimming down of the included applications and the discussion about different versions.

The workshop will be a one hour practical session on creating a personalised version of free:dyne using the base version. It will cover the creation of the LiveDVD, writing to a USB stick, setting up wireless access and personalisation. The personalisation section will include identifying and installing applications as well as changing branding and creating a user-generated ISO for distribution. A brief teaching plan of the workshop can be seen in Appendix 3. At the time of writing, a small workshop had been intended to be run with three students from the ICT class of one of the support team. However, availability of people has meant that this has had to be rescheduled. It is hoped that more can be arranged in the future.

Although on a much smaller scale, free:dyne has many of the principles outlined in the previous chapter. On an obvious level, the system is free for anyone to download and use as well as build upon and redistribute. Much like the original Linux, different versions can be forked as developers so wish. It was developed with the help of a team of distributed developers who used the internet to swap ideas, suggestions and versions. With hindsight this would have been better achieved with the use of a wiki or possibly even a blog rather than emails, however time restraints dictated otherwise. The project also had a sense of

community. Whilst it is hoped a community would be built from the construction and distribution of user-generated distros, there was also a community-feel from amongst the support team. Team members brought their own diverse skills to the project to aid its development. For example the workshop was developed in tandem with an ICT teacher who was able to organise a small working party to test the workshop. Each of the team members were enthusiastic about the project and are keen to see it succeed. The donation of their time and skills has created a gift-debt. With the responses from the team members, it does appear that they have been reciprocated with the stimulation of working on the project. But there is also one with the author who can reciprocate by acknowledging their assistance within this paper. Plus there is one owed by potential users. Without the team's skills and enthusiasm, the project would not be in the condition that it is and the potential for users to be able to create their own systems would not have been realised.

Conclusion

I may not have a lot to give but what I got I'll give to you – The Beatles

Chapter one examined the mechanisms of gift giving and the social bonds that it creates. By giving a gift there is a 'gift debt' between the giver and receiver, an obligation to reciprocate. In the hau, the gift of the birds from the forest is returned by the priests with the donating of the mauri. By performing this act, the Maori show their gratitude to the forest hoping that this will help to maintain the level of the birds for hunting. But not all forms of reciprocation are as obvious nor so immediately returned. With the Kula, a necklace is given to their neighbour on one side with a returned necklace received from the neighbour on the other side. The same with the armshells but in the opposite direction. With blood donation, the reciprocation and gift-debt seems to be with society as a whole as the giver does not expect nor demand blood to be returned from the recipient. Although Komter suggests that the reason could simply have been to be excused duties, there is more than likely a deeper motive. The giving blood is acting as an insurance, hoping that someone else has been just as generous.

Giving a gift also reveals the identities of the giver and the receiver. President Obama's gift of an iPod to the Queen certainly appears to. Whilst it might be hard to see what an eighty-three year old grandmother would want with an MP3 player, the choice of gift reinforces Obama's embracing of technology as he did with the use of online social networking sites during his campaign. The Queen's gift of a photograph of her husband and her to Obama revealed her to be more 'old school'. But this exchange also mimics that of the potlatch. In the ceremony, relations between clans are reinforced by the donation, and sometimes destruction, of wealth. Statuses are raised by not who has the most but who gives the most. Displaying that gift-giving and self-interest go hand in hand.

In the second chapter, two of the largest Open Source projects that have been instrumental in bringing awareness of OSS into the consciousness of the mainstream and even into the realms of the business world were examined. Both would not have been in the positions that they are were it not for the free donation of time and skills by hackers. With Linux, Torvalds opened the project up to the world from the very beginning. Seeking help from fellow developers paved the way for a new style of software development. By allowing

hackers to work on problems that interested them, Linux soon became a stable, robust operating system that is used widely across the Internet. This enforces Terranova's point with regard to fulfilment. In the digital economy the worker achieves fulfilment through work. With Linux, the gift of their skills was reciprocated, not through financial gain, but through the work itself. Being able to solve problems that the hackers found interesting and intellectually stimulating.

The same can be said with regard to Netscape and ultimately Firefox. However, in this case there did appear to be an element of potlatch-style competitive giving. By making the source code available for hackers to develop it further, Netscape was showing itself to be generous but there was an element of self-interest at the heart of this gesture. By freely donating its browser originally, Netscape had increased its penetration and hoped opening up the source code would do the same. With both it and Internet Explorer straying from the W3C standards, this would appear to be a good method of reining it back in by having independent developers without Netscape's influences or agenda do the work. However, the success of the community-developed Linux was not repeated in this case, at least not as quickly as had been hoped. Possibly the reason behind this was the alienation of that community Netscape has hoped to use. Instead of making use of existing licences like the GNU GPL, they created two of their own. Although there were valid business reasons behind this, it did nothing to please the hacker community. With the likes of Richard Stallman urging people not to use it, it is no wonder that the project failed or at least not be realised as quickly as Netscape had envisaged.

What can be seen from both of those examples, the use of a community for development can make or break a software project. Although on a much smaller scale it was hoped that free:dyne's development would also benefit from a community. Inspired by portable systems dyne:bolic and pure:dyne, free:dyne was purposely created to run from a USB device. This would enable the user to carry their data and their preferred choice of software with them. By simply plugging it in and booting from it, the host computer would only be used for its processing power and connection to the Internet. As with Torvalds, a request was put out for assistance, although this was to a mailing list rather than Usenet. From that, a small team of five developers were formed who were happy to give their time and skills to the project. Making use of the Internet, ideas, suggestions and software were passed around the team eventually resulting in the finished product. As with both Linux

and Firefox, free:dyne is available to download for free and can be used to create different versions that are more appropriate to the user. It is hoped in the long term that the community of developers will be enlarged to include others who have expanded free:dyne and are able to offer help and assistance.

It is undeniable that the methodology used to develop OSS has resulted in robust software. With estimates of 60,000 person-years at a cost of $8 billion to develop Linux to the current standard, it does appear to make perfect business sense to open a project up. However, that is not enough. The project must be seen to be interesting for the developers to be intellectually stimulated for them to want to freely donate their time and skills to it. As with any gift exchange there must be some form of reciprocation. Without that they will simply feel exploited.

Bibliography

Cheal, David (1988) *The Gift Economy*, London: Routledge

Gay, Joshua (ed) (2002) *Free Software, Free Society,* Sebastopol: O'Reilly

Godelier, Maurice (1999) *The Enigma of the Gift*, Cambridge: Blackwell Publishers

Himanen, Pekka (2001) *The Hacker Ethic*, New York: Random House

Hyde, Lewis (2006) *The Gift*, Edinburgh: Canongate

Kelty, Christopher M. (2008) *Two Bits The Cultural Significance of Free Software*, Durham: Duke University Press

Komter, Aafke E. (2005) *Social Solidarity and the Gift*, Cambridge: Cambridge University Press

Leadbeater, Charles (2009) *We-Think*, London: Profile Books

Levy, Stephen (2001) *Hackers Heroes of the Computer Revolution,* London: Penguin

Mauss, Marcel (1990) *The Gift*, London: Routledge

Raymond, Eric (2001) *The Cathedral and The Bazaar*, Sebastopol: O'Reilly

Schrift , Alan D. (ed) (1997) *The Logic of the Gift*, London: Routledge

Titmuss, Richard M, Ann Oakley & John Ashton (ed) (1997) *The Gift Relationship*, London: LSE

Thomas, Douglas (2002) *Hacker Culture,* Minneapolis: University of Minnesota Press

Wark, McKenzie (2004) *A Hacker Manifesto*, London: Harvard University Press

Williams, Sam (2002) *Free as in Freedom*, Sebastopol: O'Reilly

Filmography

Click BBC News 24 Originally broadcast 4th July 2009

Netography

Bill Gates' Open Letter to Hobbyists in Homebrew Club Newsletter Vol 2, Issue 1 URL: http://www.digibarn.com/collections/newsletters/homebrew/V2_01/gatesletter.html [date accessed 25th September 2009]

Cloud Computing URL: http://technology.timesonline.co.uk/tol/news/tech_and_web/article3874599.ece [date

accessed 12 August 2009]
Cloud computing is a trap, warns GNU founder Richard Stallman URL: http://www.guardian.co.uk/technology/2008/sep/29/cloud.computing.richard.stallman [date accessed 12 August 2009]
Cloud Computing URL: *http://en.wikipedia.org/wiki/Cloud_computing* [date accessed 12 August 2009]
Download Day 2008 URL: http://www.spreadfirefox.com/en-US/worldrecord/ [date accessed 16 August 2009]
dyne:bolic URL: http://dynebolic.org/ [date accessed 02 August 2009]
Kingstone, Steve (2005) *Brazil adopts open-source software* URL: http://news.bbc.co.uk/1/hi/business/4602325.stm [date accessed 18 June 2009]
Free Labor: Producing Culture for the Digital Economy - Tiziana Terranova URL: http://www.electronicbookreview.com/thread/technocapitalism/voluntary [date accessed 02 September 2009]
Microsoft's IE 6.0 is the most popular browser on the web according to OneStat.com URL: http://www.onestat.com/html/aboutus_pressbox4.html [date accessed 02 August 2009]
P2P and Human Evolution: Placing Peer to Peer Theory in an Integral Framework URL: http://www.integralworld.net/bauwens2.html [date accessed 24 June 2009]
President Obama gives the Queen an iPod URL: http://www.guardian.co.uk/music/2009/apr/02/barack-obama-presents-queen-ipod [date accessed 12 September 2009]
The Gift Economy URL: http://www.context.org/ICLIB/IC41/PinchotG.htm [date accessed 01 August 2009]
The Gift Economy and Free Software URL: http://www.linux.com/archive/articles/36554
Live Aid URL: http://www.bobgeldof.info/Charity/liveaid.html [date accessed 18 July 2009]
The Open Source Definition URL: http://www.opensource.org/docs/osd [date accessed 09 August 2009]
Zawinski, Jamie (1999) *nomo zilla* URL: http://www.jwz.org/gruntle/nomo.html [date accessed 21 July 2009]

Conferences

FAVE 2005, Trinity Community & Arts Centre, Bristol 20 August 2005

Apollo 33, Ride Cafe, Plymouth 16 November 2005

FAVE 2006 Limehouse Town Hall, London 25 November 2006

Appendices

Appendix 1

Initial email

From: On Behalf Of chris saunders
Sent: 05 July 2009 2:52 PM
To: FN-FORUM
Subject: FN-FORUM: Linux question
Hi everyone
I'm about to start a project that will involve having a bootable system
on a usb stick and I was wondering if anyone had any suggestions as to
which version of Linux to use. I'm thinking of using Ubuntu 8.10 but I'm
open to suggestions.

Thanks

Chris

Replies

From: On Behalf Of David Clough
Sent: 05 July 2009 3:53 PM
To: FN-FORUM
Subject: RE: FN-FORUM: Linux question
9.04... it's amazing, far better than 8 in so many ways...

From: On Behalf of Peter Lewis
Sent: 06 July 2009 10:02 AM
To: FN-FORUM
Subject: RE: FN-FORUM: Linux question
Hi Chris
Have to agree 9.04 has a lot more features in it depending on the nature of your project.
OpenSUSE is worth a look to

Regards,
Peter

From: On Behalf of Heard, Gerry
Sent: 06 July 2009 12:36 PM
To: FN-FORUM
Subject: RE: FN-FORUM: Linux question
hello chris

ive used 8.10 and its fairly robust. 9.04 looks good but im yet to upgrade to it. Be keen to hear how you get on
cheers gerry

From: On Behalf of nic dacres
Sent: 07 July 2009 7:17 PM
To: FN-FORUM
Subject: RE: FN-FORUM: Linux question
9.04 is well worth a look
nic

From: On Behalf of Alex Beston
Sent: 18 July 2009 8:43 AM
To: FN-FORUM
Subject: RE: FN-FORUM: Linux question
Hello

I would definitely use 9.04. It has a sweet built in feature for writing itself to a usb stick.

Regards,
Al

Appendix 2

List of applications

Graphics

GIMP

Inkscape

Office

Dictionary

Evolution mail & calendar

OpenOffice

Project

Programming

Arduino

gPHPEdit

Kompozer

Netbeans

Processing

Python

Audio & Visual

Audacity

Brasero Disc Burner

DJ Play

Jack Control

Kino

Media Player

Mixx

Pure Data

Rhythm Box

Sound Recorder

VLC

Appendix 3

Workshop Format

Introduction
Talk about self and project

Create installation
Handout copies of LiveDVD, boot up, run thru 'copying to USB device run thru 'BIOS changes, boot from USB

Using
Connecting to wireless network, quick run thru' applications

Personalising installation
Changing background, installing/removing applications

Creating distro
Remastersys